Randomized Chess

Lestyán Ádám Barnabás

FO6K58

lestyan.adam.01@gmail.com

Felhasználói Kézikönyv

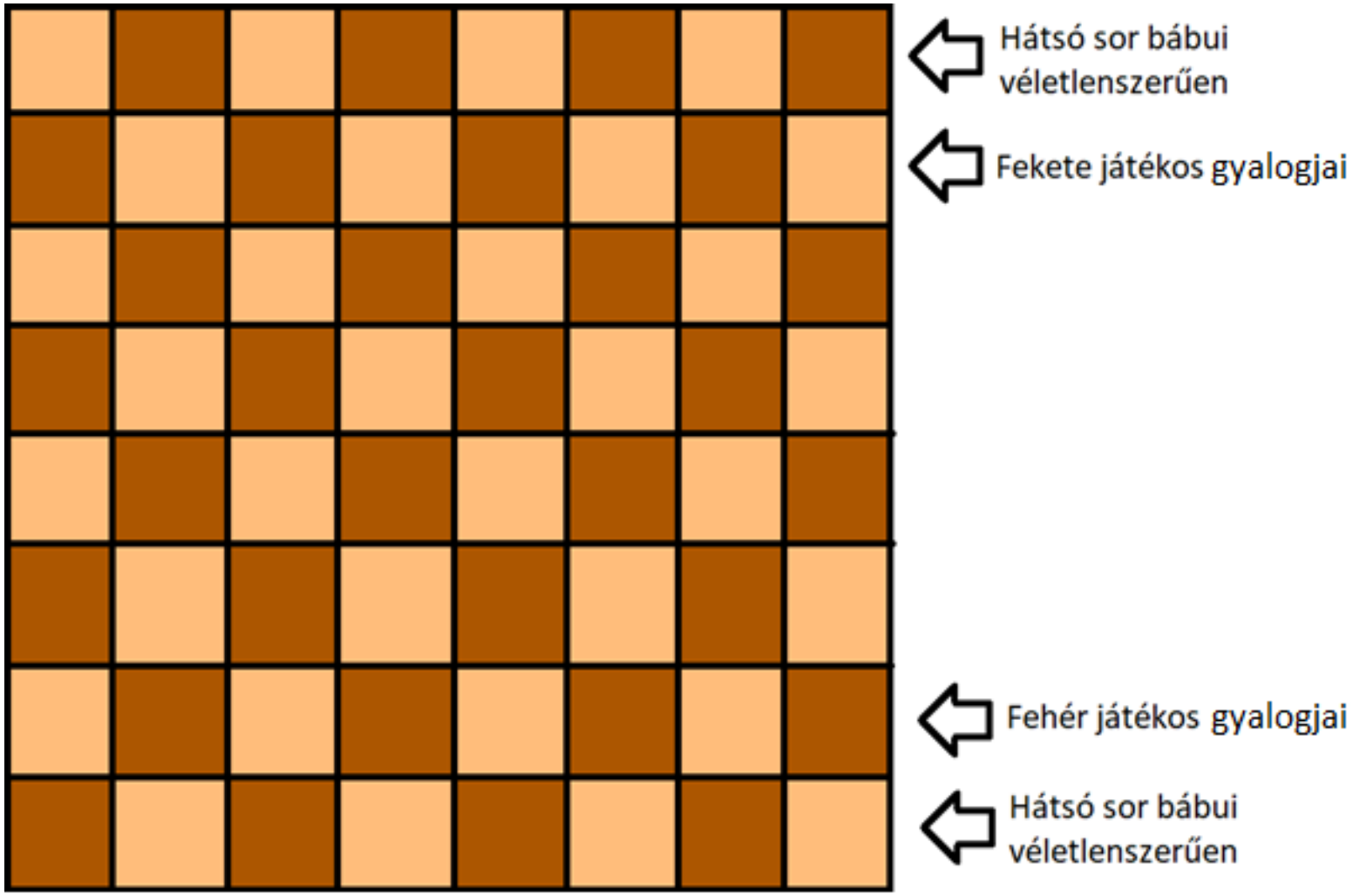
1. Játék leírása, szabályai

A játék egy sakk, amelyben a hátsó sorokban véletlenszerűen helyezzük el a bábukat.A játék két személyes és egy 8x8 -as táblán játsszák. Mindkét játékosnak (fehér és fekete) van 16 bábuja, amiből 8 gyalog, 2 bástya, 2 futó, 2 huszár, 1 királynő és 1 király. A játékosok felváltva lépnek (elsőként a fehér kezd) egyszerre egy bábuval. Mindegyik fajta bábu máshogy tud lépni. Egy mezőn egyszerre csak egy bábu állhat. Ha egy mezőn az ellenfél bábuja áll, akkor azt le lehet ütni, ekkor az ellenfél figuráját levesszük a tábláról, és saját, odalépő bábunkat tesszük a helyére. A cél az ellenfél királyának leütése.

A bábuk:

Balról jobbra: Király, Királynő, Futó, Huszár, Bástya, Gyalog

A kiindulási helyzet:
Mindkét oldalon az utolsó előtti sorokban (2. és 7.) helyezzük el a játékos gyalogjait. A fennmaradó bábukat véletlenszerűen helyezzük el az utolsó sorokban (1. és 8.).



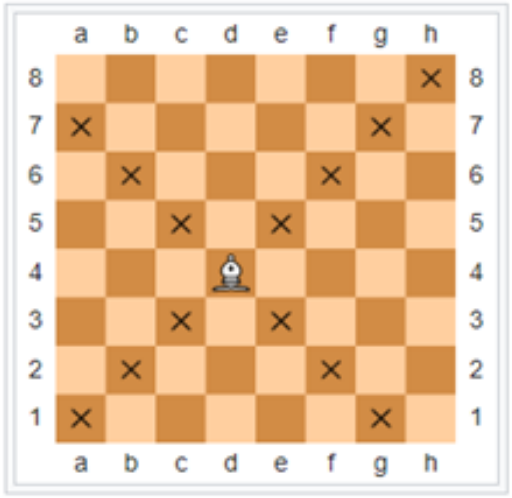
A bábuk lépései:

Mindegyik fajta bábu, kivéve a gyalogot, leütheti az ellenfél bábuját, ha az olyan helyen áll, ahova mozoghat. A gyalog csak előre léphet egyet, vagy kettőt, ha még nem léptek vele, viszont csak átlósan ütheti le az ellenfél bábuját. Az alábbi ábrákon látható, hogy melyik bábu, hogy léphet.

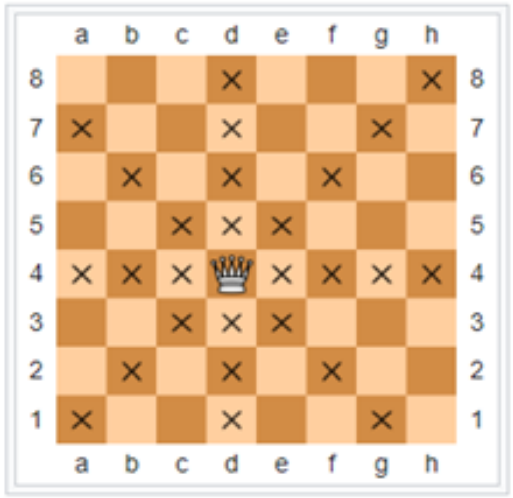
Király

Bástya

Futó



Királynő



Huszár

Gyalog

2. Felhasználói felület kezelése

A játék indítása után a start menü fogadja a felhasználót. A start menüben 3 opció közül lehet választani:

- New Game – Új Játék indítása
- Previous Game – Az előző játék lépéseinek megtekintése
- Exit – Kilépés a programból

Új Játék

A New Game opció kiválasztásakor bezáródik a start menü és megjelenik a játék ablaka, amiben játszani lehet. Az ablakban egy 8x8 -as sakktábla jelenik meg, rajta a bábukkal.
Az alsó két sorban a fehér játékos bábui vannak elhelyezve, a 2. sorban a gyalogok, míg az 1. sorban a maradék bábuk véletlenszerűen elhelyezve. Az egyetlen kikötés az, hogy a futók különböző színű mezőkön legyenek, ettől eltekintve teljesen véletlenszerűen vannak elhelyezve a bábuk. A felső két sorban a fekete bábuk vannak elhelyezve hasonló módon.

Lépés

A bábukkal való lépés a következőképpen történik:
A felhasználó rákattint egy bábura és a bábu ezáltal ki lesz választva. Az aktuálisan kiválasztott bábu pirosan be van keretezve. Miután egy bábu ki lett választva nem lehet más bábut kiválasztani, tehát kiválasztás után muszáj azzal a bábuval lépni. A felhasználó nem tud olyan bábut kiválasztani, amivel nem lehet lépni.
A játék kezdetekor elsőként a fehér játékos lép, aztán következik a fekete játékos és felváltva lépnek, addig amíg vége nem lesz a játéknak. Egy játékos nem tudja kiválasztani az ellenfél bábuját.
Egy bábu kiválasztása után a játékos rákattint arra a mezőre, amire lépni szeretne.
Amennyiben a mezőn, amire rákattintott, volt az ellenfélnek bábuja, akkor az a bábu le lett ütve és lekerül a tábláról és a helyére a játékos bábuja kerül.
A bábu nem tud ugyanarra a mezőre lépni, amin eddig tartózkodott, csak olyan mezőre léphet, amelyre a lépései engedik.

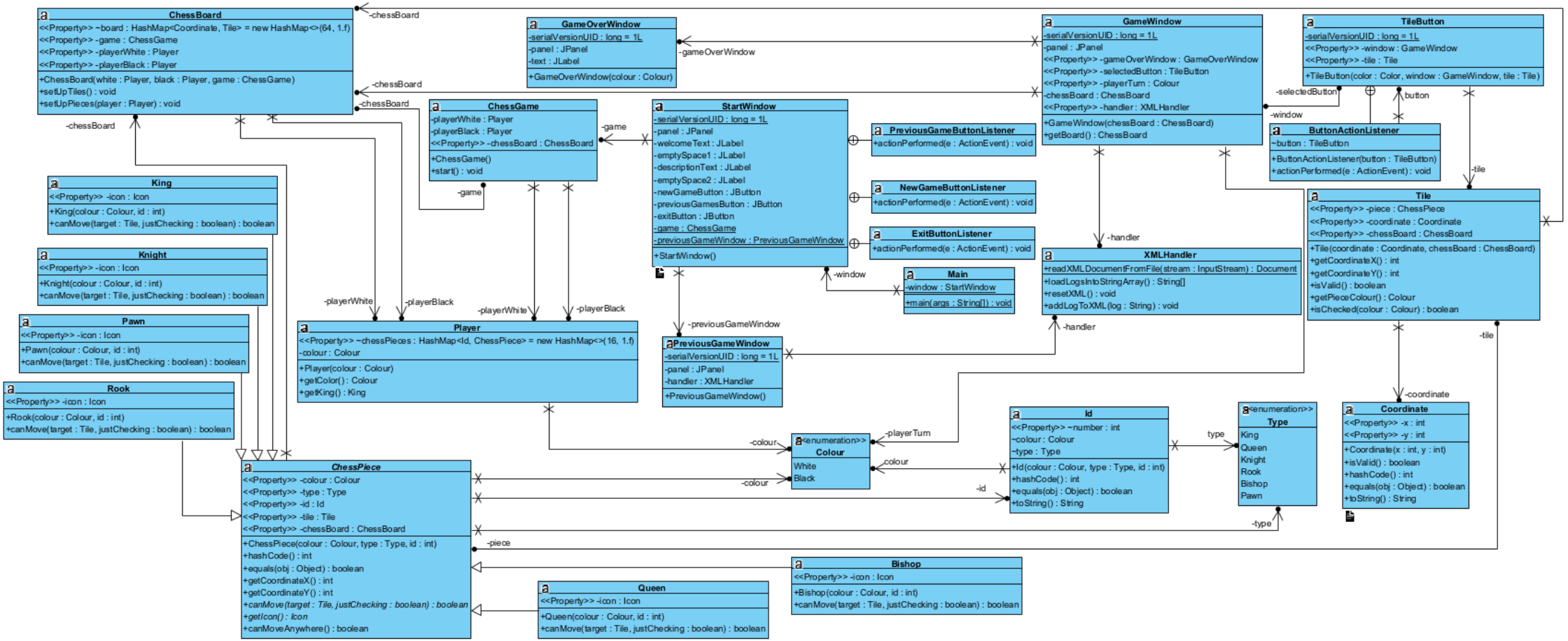
Sakk
Sakknak nevezik azt az esetet, amikor az egyik játékos bábuja le tudná ütni a következő körben az ellenfél királyát. Abban az esetben, amikor az ellenfél sakkban tartja a játékos királyát, a játékos csak a királyát tudja kiválasztani, muszáj azzal lépnie. Ha a játékos nem tud a királlyal lépni, akkor vesztett. Ezt nevezik sakk-matt-nak.
Egy játékos csak olyan mezőre léphet a királyával, amire, ha lépne nem lenne sakkban.

Game Over
Sakk-matt esetén vége van a játéknak és bezáródik a játék ablaka és a program egy új ablakban kiírja, hogy melyik játékos győzött.

Előző Játék
A start menüben a Previous Game opció választásakor a start menü bezáródik és egy új menüben megjelenik az előző játék lépéseinek listája vagy, ha nem volt előző játék, akkor egy üzenet, hogy nincs mit megjeleníteni.

Kilépés
A start menüben az Exit opció választásakor a program bezáródik.

UML Class Diagram



Metódusok ismertetése

ChessPiece

- public ChessPiece(Colour, Type, int)
  - A ChessPiece osztály konstruktora. Egy színből, egy típusból és egy számból létrehoz egy ChessPiece objektumot, és közben meghívja az Id osztály konstruktorát és létrehoz egy Id objektumot a ChessPiece – nek.
- public Colour getColour()
  - Visszaadja visszatérési értékként a ChessPiece colour adattagját.
- public void setColour(Colour)
  - Beállítja a ChessPiece colour adattagját a paraméterként megadott színre.
- public Type getType()
  - Visszaadja visszatérési értékként a ChessPiece type adattagját.
- public int hashCode()
  - Felülírja az alapértelmezett hashelő függvényt, hogy csak az adattagokat hashelje.
- public boolean equals(Object)
  - Felülírja az alapértelmezett equals függvényt és ezáltal lehetővé teszi, hogy érték szerint hasonlítsunk össze ChessPiece – eket.
- public int getId()
  - Visszaadja visszatérési értékként a ChessPiece Id adattagját.
- public int getCoordinateX()
  - Visszaadja visszatérési értékként annak a mezőnek az x koordinátáját, amin a ChessPiece áll.
- public int getCoordinateY()
  - Visszaadja visszatérési értékként annak a mezőnek az y koordinátáját, amin a ChessPiece áll.

- public Tile getTile()
  - Visszaadja visszatérési értékként azt a mezőt, amin a ChessPiece áll.
- public void setTile(Tile)
  - Beállítja a ChessPiece objektumohz tartozó mezőt a paraméterként megadott mezőre.
- public ChessBoard getChessBoard()
  - Visszaadja visszatérési értékként azt a sakktáblát, amihez hozzá van rendelve a bábu.
- public void setChessBoard(ChessBoard)
  - Beállítja a ChessPiece chessBoard adattagját a paraméterként megadott sakktáblára.
- public abstract boolean canMove(Tile, boolean)
  - Ez egy absztrakt függvény, ami a leszármazottakban, ha megvalósításra kerül, akkor megállapítja egy mezőről, hogy oda tud-e lépni a bábu. Amennyiben oda tud lépni, akkor igaz a visszatérési érték, ellenkező esetben hamis. A boolean paraméter, abban az esetben igaz, ha az isChecked függvény alatt kerül meghívásra. Ebben az esetben úgy kell kezelni a megadott mezőt, mintha azon állna az ellenfél királya.
- public abstract Icon getIcon()
  - Visszaadja visszatérési értékként a bábu ikonját, ha meg van valósítvaa függvény.
- public boolean canMoveAnywhere()
  - Ez a tagfüggvény megállapítja a báburól, hogy tud-e valamerre lépni. Abban az esetben, ha egy mezőre is képes lépni a bábu, akkor igazat ad vissza a függvény, ellenkező esetben pedig hamisat.

Bishop

- public Bishop(Colour, int)
  - A Bishop osztály konstruktora, egy színből és egy számból készít egy futót és betölti a futó ikonját.
- public boolean canMove(Tile, boolean)
  - Megállapítja egy mezőről, hogy oda tud-e lépni a futó. Amennyiben oda tud lépni, akkor igaz a visszatérési érték, ellenkező esetben hamis. A boolean paraméter, abban az esetben igaz, ha az isChecked függvény alatt

kerül meghívásra. Ebben az esetben úgy kell kezelni a megadott mezőt, mintha azon állna az ellenfél királya.

- public Icon getIcon()
  - Visszaadja visszatérési értékként a futó ikonját.

Knight

- public Knight(Colour, int)
  - A Knight osztály konstruktora, egy színből és egy számból készít egy huszárt és betölti a huszár ikonját.
- public boolean canMove(Tile, boolean)
  - Megállapítja egy mezőről, hogy oda tud-e lépni a huszár. Amennyiben oda tud lépni, akkor igaz a visszatérési érték, ellenkező esetben hamis. A boolean paraméter, abban az esetben igaz, ha az isChecked függvény alatt kerül meghívásra. Ebben az esetben úgy kell kezelni a megadott mezőt, mintha azon állna az ellenfél királya.
- public Icon getIcon()
  - Visszaadja visszatérési értékként a huszár ikonját.

Queen

- public Queen(Colour, int)
  - A Queen osztály konstruktora, egy színből és egy számból készít egy királynőt és betölti a királynő ikonját.
- public boolean canMove(Tile, boolean)
  - Megállapítja egy mezőről, hogy oda tud-e lépni a királynő. Amennyiben oda tud lépni, akkor igaz a visszatérési érték, ellenkező esetben hamis. A boolean paraméter, abban az esetben igaz, ha az isChecked függvény alatt kerül meghívásra. Ebben az esetben úgy kell kezelni a megadott mezőt, mintha azon állna az ellenfél királya.

- public Icon getIcon()
    - Visszaadja visszatérési értékként a királynő ikonját.

Rook

- public Rook(Colour, int)
    - A Rook osztály konstruktora, egy színből és egy számból készít egy bástyát és betölti a bástya ikonját.
- public boolean canMove(Tile, boolean)
    - Megállapítja egy mezőről, hogy oda tud-e lépni a bástya. Amennyiben oda tud lépni, akkor igaz a visszatérési érték, ellenkező esetben hamis. A boolean paraméter, abban az esetben igaz, ha az isChecked függvény alatt kerül meghívásra. Ebben az esetben úgy kell kezelni a megadott mezőt, mintha azon állna az ellenfél királya.
- public Icon getIcon()
    - Visszaadja visszatérési értékként a bástya ikonját.

King

- public King(Colour, int)
    - A King osztály konstruktora, egy színből és egy számból készít egy királyt és betölti a király ikonját.
- public boolean canMove(Tile, boolean)
    - Megállapítja egy mezőről, hogy oda tud-e lépni a király. Amennyiben oda tud lépni, akkor igaz a visszatérési érték, ellenkező esetben hamis. A boolean paraméter, abban az esetben igaz, ha az isChecked függvény alatt kerül meghívásra. Ebben az esetben úgy kell kezelni a megadott mezőt, mintha azon állna az ellenfél királya.
- public Icon getIcon()
    - Visszaadja visszatérési értékként a király ikonját.

Pawn

- public Pawn(Colour, int)
  - A Pawn osztály konstruktora, egy színből és egy számból készít egy gyalogot és betölti a gyalog ikonját.
- public boolean canMove(Tile, boolean)
  - Megállapítja egy mezőről, hogy oda tud-e lépni a gyalog. Amennyiben oda tud lépni, akkor igaz a visszatérési érték, ellenkező esetben hamis. A boolean paraméter, abban az esetben igaz, ha az isChecked függvény alatt kerül meghívásra. Ebben az esetben úgy kell kezelni a megadott mezőt, mintha azon állna az ellenfél királya.
- public Icon getIcon()
  - Visszaadja visszatérési értékként a gyalog ikonját.

Id

- public Id(Colour, Type, int)
  - Az Id osztály konstruktora. Létrehoz egy Id – t egy színből, egy típusból és egy számból.
- public int getNumber()
  - Visszaadja visszatérési értékként az Id number adattagját.
- public int hashCode()
  - Felülírja az alapértelmezett hashelő függvényt, hogy csak az adattagokat hashelje.
- public boolean equals(Object)
  - Felülírja az alapértelmezett equals függvényt és ezáltal lehetővé teszi, hogy érték szerint hasonlítsunk össze Id – kat.
- public String toString()
  - Felülírja az alapértelmezett toString metódust. Visszaad visszatérési értékként egy string-et, amivel ki tudunk iratni egy Id objektumot.

Coordinate

- public Coordinate(int, int)
  - A Coordinate osztály konstruktora. Létrehoz egy Coordinate objektumot két számból.
- public int getX()
  - Visszaadja visszatérési értékként a koordináta x adattagját.
- public int getY()
  - Visszaadja visszatérési értékként a koordináta y adattagját.
- public boolean isValid()
  - Megállapítja a koordinátáról, hogy helyes-e. Egy 8x8-as tábla esetén ez azt jelenti, hogy az x és y koordinátáknak 1 és 8 között kell lenniük. Ha helyesek a koordináták igazat adunk vissza, ha nem akkor hamisat.
- public int hashCode()
  - Felülírja az alapértelmezett hashelő függvényt, hogy csak az adattagokat hashelje.
- public boolean equals(Object)
  - Felülírja az alapértelmezett equals függvényt és ezáltal lehetővé teszi, hogy érték szerint hasonlítsunk össze Coordinate - ket.
- public String toString()
  - Felülírja az alapértelmezett toString metódust. Visszaad visszatérési értékként egy string-et, amivel ki tudunk iratni egy Coordinate objektumot.

Player

- public Player(Colour)
  - A Player osztály konstruktora. Létrehoz egy Player objektumot egy színből és feltölti a Player HashMap – jét bábukkal.
- public Colour getColor()
  - Visszaadja visszatérési értékként a Player színét.
- public King getKing()
  - Visszaadja visszatérési értékként a Player HashMap - jéből a királyt.
- public HashMap<Id , ChessPiece> getChessPieces()
  - Visszaadja visszatérési értékként a Player Id – bábu párokat tartalmazó HashMap – jét.

ChessBoard

- public ChessBoard(Player, Player, ChessGame)
  - A ChessBoard osztály konstruktora. Létrehoz egy ChessBoard objektumot két Playerből és egy ChessGame -ből.
- public void setUpTiles()
  - Létrehoz Coordinate – Tile párokat és feltölti velük a sakktábla HashMap -jét.
- public void setUpPieces(Player)
  - Elhelyezi a paraméterként kapott játékos bábuit a sakktáblán. Ha a játékos fehér, akkor a gyalogjait elhelyezi a 2. sorban és a maradék bábuit az 1. sorban, azzal a kikötéssel, hogy a futók különböző színeken kell, hogy álljanak. Ha a játékos fekete, akkor a bábukat ugyanígy helyezi el a 7. és 8. sorokban.
- public Player getPlayerWhite()
  - Visszaadja visszatérési értékként a fehér játékost.

- public Player getPlayerBlack()
  - Visszaadja visszatérési értékként a fekete játékost.
- public ChessGame getGame()
  - Visszaadja visszatérési értékként a sakktábla ChessGame adattagját.
- public HashMap<Coordinate, Tile> getBoard()
  - Visszaadja visszatérési értékként a sakktábla HashMap -jét.

Tile

- public Tile(Coordinate, ChessBoard)
  - A Tile osztály konstruktora. Létrehoz egy Tile objektumot egy Coordinate és egy ChessBoard -ból.
- public ChessPiece getPiece()
  - Visszaadja visszatérési értékként az ezen a mezőn álló sakk bábut.
- public void setPiece(ChessPiece)
  - Beállítja a mezőhőz tartozó bábut a paraméterként kapott bábura.
- public Coordinate getCoordinate()
  - Visszaadja visszatérési értékként a mező koordinátáját.
- public int getCoordinateX()
  - Visszaadja visszatérési értékként a mező x koordinátáját.
- public int getCoordinateY()
  - Visszaadja visszatérési értékként a mező x koordinátáját.
- public boolean isValid()
  - Megállapítja a mező koordinátájáról, hogy helyes-e. Ha helyes a koordináta igazat adunk vissza, ha nem akkor hamisat.
- public Colour getPieceColour()
  - Visszaadja visszatérési értékként a mezőn álló bábu színét.

- public boolean isChecked(Colour)
    - Megállapítja egy mezőről, hogy támadja-e a paraméterként kapott színnel ellentétes játékos bábui közül bármelyik. Ha valamelyik bábu le tudná ütni az ezen a mezőn tartózkodó bábut, akkor igazat ad vissza, ellentétes esetben hamisat.
- public ChessBoard getChessBoard()
    - Visszaadja visszatérési értékként azt a sakktáblát, amelyikhez a mező tartozik.

ChessGame

- public ChessGame()
    - A ChessGame osztály konstruktora. Létrehozza a játékosokat és a sakktáblát.
- public void start()
    - Elindítja a játékot, azaz létrehozza és megjeleníti a játék ablakát.
- public ChessBoard getChessBoard()
    - Visszaadja visszatérési értékként a játékhoz tartozó sakktáblát.

TileButton

- public TileButton(Color, GameWindow, Tile)
    - A TileButton osztály konstrukora. Létrehoz egy a paraméterként megkapott színű JButton-t és hozzárendeli egy mezőhöz és egy játék ablakhoz.
- public GameWindow getWindow()
    - Visszaadja visszatérési értékként azt a játék ablakot, amelyhez a gomb tartozik.
- public Tile getTile()
    - Visszaadja visszatérési értékként azt a mezőt, amelyhez a gomb tartozik.

ButtonActionListener

- public ButtonActionListener(TileButton)
  - Az ButtonActionListener privát osztály konstruktora. Létrehoz egy ButtonActionListener objektumot a paraméterként kapott gombhoz.
- public void actionPerformed(ActionEvent)
  - Ezen a függvényen belül valósul meg a játék logikája. Egy TileButton megnyomásakor hívódik meg a függvény és eldönti, hogy ha még nem volt kiválasztva gomb, akkor ki lehet-e választani a megnyomott gombot, vagy ha már volt kiválasztott gomb, akkor a kiválasztott gomb mezőjén lévő bábuval oda lehet-e lépni a megnyomott gomb mezőjére.

StartWindow

- public StartWindow()
  - A StartWindow osztály konstruktora. Létrehozza a start menüt megjelenítő ablakot.

NewGameButtonListener

- actionPerformed(ActionEvent)
  - A New Game gomb megnyomására hívódik meg és bezárja a start menü ablakát, illetve létrehoz és elindít egy új játékot.

PreviousGameButtonListener

- actionPerformed(ActionEvent)
  - A Previous Game gomb megnyomására hívódik meg és bezárja a start menü ablakát, illetve létrehozza és megjeleníti az előző játék lépéseit megjelenítő ablakot.

ExitButtonListener

- actionPerformed(ActionEvent)
  - Az Exit gomb megynomására hívódik meg és bezárja a start menü ablakát és leállítja a programot.

PreviousGameWindow

- public PreviousGameWindow()
  - A PreviousGameWindow osztály konstruktora. Létrehoz egy ablakot, beolvassa az előző játék lépéseit és megjeleníti őket az ablakban. Ha nem volt előző játék vagy nem léptek az előző játékban akkor egy erről szóló üzenetet jelenít meg az ablakban.

GameOverWindow

- public GameOverWindow(Colour)
  - A GameOverWindow osztály konstruktora. Létrehoz egy ablakot és az ablakban tájékoztatja a felhasználót arról, hogy a paraméterként kapott színű játékos nyerte meg a játékot.

GameWindow

- public GameWindow(ChessBoard)
  - A GameWindow osztály konstruktora. Létrehozza a játék ablakát és a paraméterként kapott sakktábla mezőihez tartozó gombokat.
- public ChessBoard getBoard()
  - Visszaadja visszatérési értékként a játék ablakhoz tartozó sakktáblát.
- public TileButton getSelectedButton()
  - Visszaadja visszatérési értékként azt a gombot, amelyik ki van választva vagy null-t ha nincs még gomb kiválasztva.

- public void setSelectedButton(TileButton)
  - Beállítja a paraméterként kapott gombot a kiválasztott gombnak.
- public Colour getPlayerTurn()
  - Visszaadja visszatérési értékként azt a színt, amely azé a játékosé, amelyik éppen következik.
- public void setPlayerTurn(Colour)
  - Beállítja a jelenleg következő játékos színét a paraméterként kapott színre. Nem változtatja meg egy játékosnak sem a színét, csak a GameWindow privát szín típusú változóját frissíti.
- public XMLHandler getHandler()
  - Visszaadja visszatérési értékként azt az XMLHandler objektumot, amelyik az ablakhoz tartozik.
- public GameOverWindow getGameOverWindow()
  - Visszaadja visszatérési értékként azt a GameOverWindow objektumot, amely a GameWindow-hoz tartozik.
- public void setGameOverWindow(GameOverWindow)
  - Beállítja a GameWindow – hoz tartozó GameOverWindow objektumot arra a GameOverWindow objektumra, amelyet paraméterként kap.

XMLHandler

- public static Document readXMLDocumentFromFile(InputStream)
  - A paraméterként kapott InputStream-ből készít egy dokumentumot és visszaadja visszatérési értékként.
- public String[] loadLogsIntoStringArray()
  - A projekthez tartozó xml fájlból beolvassa az előző játék lépéseit egy String tömbbe és visszaadja visszatérési értékként.
- public void resetXML()
  - A projekthez tartozó xml fájlból kitörli az összes rögzített lépést.
- public void addLogToXML(String)
  - A paraméterként kapott Stringet hozzáadja a projekthez tartozó xml fájlhoz.

Tesztek rövid leírása

PlayerTest

- A PlayerTest teszt osztály teszteli, hogy a Player osztály konstruktora létrehozza-e az összes bábut helyesen.

ChessBoardTest

- A ChessBoardTest teszt osztály teszteli, hogy a ChessBoard osztály konstruktora létrehozza-e az összes mezőt helyesen és helyesen helyezi-e el a két játékos bábuit.

IsCheckedTest

- Az IsCheckedTest teszt osztály teszteli, hogy helyesen működik-e a Tile osztály isChecked metódusa

CanMoveAnywhereTest

- A CanMoveAnywhereTest teszt osztály teszteli, hogy helyesen működik-e a ChessPiece osztály canMoveAnywhere metódusa.

BishopMovesTest

- A BishopMovesTest teszt osztály teszteli, hogy helyesen lép-e a futó.

PawnMovesTest

- A PawnMovesTest teszt osztály teszteli, hogy helyesen lép-e a gyalog.

RookMovesTest

- A RookMovesTest teszt osztály teszteli, hogy helyesen lép-e a bástya.

KnightMovesTest

- A KnightMovesTest teszt osztály teszteli, hogy helyesen lép-e a huszár.

QueenMovesTest

- A QueenMovesTest teszt osztály teszteli, hogy helyesen lép-e a királynő.

KingMovesTest

- A KingMovesTest teszt osztály teszteli, hogy helyesen lép-e a király.